REPORT DOCUMENTATION PAGE

Form Approved OMB NO. 0704-0188

1. REPORT DATE (DD-MM-YYYY) 18-08-2014	2. REPORT TYPE Final Report	3. DATES COVERED (From - To) 1-May-2013 - 30-Apr-2016

4. TITLE AND SUBTITLE Final Report: Advanced Overset Grid Methods for Massively Parallel Rotary Wing Computations	5a. CONTRACT NUMBER W911NF-13-1-0150
	5b. GRANT NUMBER
	5c. PROGRAM ELEMENT NUMBER 611102
6. AUTHORS Jayanarayanan Sitaraman	5d. PROJECT NUMBER
	5e. TASK NUMBER
	5f. WORK UNIT NUMBER
7. PERFORMING ORGANIZATION NAMES AND ADDRESSES University of Wyoming Box 3295 University Station Laramie, WY 82071 -3295	8. PERFORMING ORGANIZATION REPORT NUMBER
9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS (ES) U.S. Army Research Office P.O. Box 12211 Research Triangle Park, NC 27709-2211	10. SPONSOR/MONITOR'S ACRONYM(S) ARO
	11. SPONSOR/MONITOR'S REPORT NUMBER(S) 62547-EG.1

12. DISTRIBUTION AVAILIBILITY STATEMENT

Approved for Public Release; Distribution Unlimited

13. SUPPLEMENTARY NOTES

The views, opinions and/or findings contained in this report are those of the author(s) and should not contrued as an official Department of the Army position, policy or decision, unless so designated by other documentation.

14. ABSTRACT

Parallel overset grid assembly techniques have recently been implemented in the U.S. Army Helios software for rotorcraft aeromechanics simulations. The work presented in this report describes an innovative active load balancing algorithm that improves the robustness and scalability of domain connectivity operations. Another aspect of this works in to development of conservative overset grid methods. Preliminary results in 2-D for cell-centered unstructured grid show feasibility of this approach and improvements in predictions when compared with

15. SUBJECT TERMS

Comptuational Fluid Dynamics, Overset Grids, Parallel Computing

16. SECURITY CLASSIFICATION OF:			17. LIMITATION OF ABSTRACT	15. NUMBER OF PAGES	19a. NAME OF RESPONSIBLE PERSON
a. REPORT	b. ABSTRACT	c. THIS PAGE			Jayanarayanan Sitaraman
UU	UU	UU	UU		19b. TELEPHONE NUMBER 307-766-5424

Report Title

Final Report: Advanced Overset Grid Methods for Massively Parallel Rotary Wing Computations

ABSTRACT

Parallel overset grid assembly techniques have recently been implemented in the U.S. Army Helios software for rotorcraft aeromechanics simulations. The work presented in this report describes an innovative active load balancing algorithm that improves the robustness and scalability of domain connectivity operations. Another aspect of this works in to development of conservative overset grid methods. Preliminary results in 2-D for cell-centered unstructured grid show feasibility of this approach and improvements in predictions when compared with
conventional overset grid approaches.

Enter List of papers submitted or published that acknowledge ARO support from the start of the project to the date of this printing. List the papers, including journal references, in the following categories:

(a) Papers published in peer-reviewed journals (N/A for none)

Received Paper

TOTAL:

Number of Papers published in peer-reviewed journals:

(b) Papers published in non-peer-reviewed journals (N/A for none)

Received Paper

TOTAL:

Number of Papers published in non peer-reviewed journals:

(c) Presentations

Number of Presentations: 0.00

Non Peer-Reviewed Conference Proceeding publications (other than abstracts):

Received	Paper

TOTAL:

Number of Non Peer-Reviewed Conference Proceeding publications (other than abstracts):

Peer-Reviewed Conference Proceeding publications (other than abstracts):

Received	Paper

TOTAL:

Number of Peer-Reviewed Conference Proceeding publications (other than abstracts):

(d) Manuscripts

Received	Paper

TOTAL:

Number of Manuscripts:

Books

Received	Book

TOTAL:

Received	Book Chapter

TOTAL:

Patents Submitted

Patents Awarded

Awards

Graduate Students

NAME	PERCENT_SUPPORTED	Discipline
Orhan Shibliyev	1.00	
Phillip Davidson	0.50	
FTE Equivalent:	**1.50**	
Total Number:	**2**	

Names of Post Doctorates

NAME	PERCENT_SUPPORTED
FTE Equivalent:	
Total Number:	

Names of Faculty Supported

NAME	PERCENT_SUPPORTED	National Academy Member
Jayanarayanan Sitaraman	0.17	
Beatrice Roget	0.50	
FTE Equivalent:	**0.67**	
Total Number:	**2**	

Names of Under Graduate students supported

NAME	PERCENT_SUPPORTED
FTE Equivalent:	
Total Number:	

Student Metrics

This section only applies to graduating undergraduates supported by this agreement in this reporting period

The number of undergraduates funded by this agreement who graduated during this period:	0.00
The number of undergraduates funded by this agreement who graduated during this period with a degree in science, mathematics, engineering, or technology fields:......	0.00
The number of undergraduates funded by your agreement who graduated during this period and will continue to pursue a graduate or Ph.D. degree in science, mathematics, engineering, or technology fields:......	0.00
Number of graduating undergraduates who achieved a 3.5 GPA to 4.0 (4.0 max scale):......	0.00
Number of graduating undergraduates funded by a DoD funded Center of Excellence grant for Education, Research and Engineering:......	0.00
The number of undergraduates funded by your agreement who graduated during this period and intend to work for the Department of Defense	0.00
The number of undergraduates funded by your agreement who graduated during this period and will receive scholarships or fellowships for further studies in science, mathematics, engineering or technology fields:......	0.00

Names of Personnel receiving masters degrees

NAME

Total Number:

Names of personnel receiving PHDs

NAME

Total Number:

Names of other research staff

NAME	PERCENT_SUPPORTED

FTE Equivalent:
Total Number:

Sub Contractors (DD882)

Inventions (DD882)

Scientific Progress

See attachment

Technology Transfer

Close interaction with Helios development team at Army Aeroflightdynamics directorate at NASA Ames

Advanced Overset Grid Methods For Massively Parallel Rotary Wing Computations
Final Report

Jayanarayanan Sitaraman[1]

[1]Department of Mechanical Engineering, University of Wyoming, Laramie, WY 82071, United States

Contents

Chapter 1

Introduction

The use of Computational Fluid Dynamics (CFD) to simulate certain complex problems remains challenging, for example when considering the flow around multiple moving bodies, deforming bodies, or bodies with very complex geometries. This is because the traditional approach of using a single grid to discretize the flow domain would require complex and problem-dependent operations such as grid stretching and re-meshing. The overset (or Chimera) grid approach provides an elegant solution to this problem by allowing the use of several overlapping meshes to discretize different parts of the flow domain. However, the presence of overlap requires additional tasks to be performed, such as determining whether the solution should be computed or interpolated at each grid point, and which grid points (such as inside a solid body) should be ignored by the solver. This problem as a whole is termed Overset Grid Assembly (OGA). Its solution can be quite challenging, particularly when considering multiple unstructured meshes partitioned and distributed among several processors. Point containment search, or donor search, which involves determining the cell (or cells) that encompass a given query point, is the core algorithm that is required for the overset grid assembly process. One of the prominent challenges, for partitioned meshes, is to design a robust donor search algorithm that can account for the the complex geometry that is often generated at the boundary of each mesh-block during partitioning. One of the most efficient algorithms for donor search utilizes a line-walk process (variants are also called stencil-walk or stencil-jump). In general, this algorithm entails selecting a seed cell in the neighborhood of the given query point and then "walking" towards the point across the cell boundaries using the connectivity data of cell neighbors. In Figure 1.1 a typical unstructured grid partition is shown where the complex nature of the partition boundary can be clearly observed. Further, it is illustrated that the line walking algorithm in its traditional form will incur robustness issues because of multiple exits and reentry of the search-line through the partition boundary. One way to tackle this problem is to continue the search-line across partitions and transfer control of the donor search to another processor that controls the neighboring unstructured partition. However, such process would create asynchronous parallel communication patterns that will affect the overall efficiency of the algorithm. To enable the simulation of unsteady flow problems with relative motion and elasticity, the overset grid assembly needs to be performed at each time step. Therefore, it is critical for the assembly process to be as efficient as possible. In this context, the primary focus of this paper is to develop a donor search algorithm that is efficient, uses only local data, and is sufficiently robust for handling the complex geometrical nature of the partition boundaries.

Since the problem of Overset Grid Assembly was first introduced in the early eighties [23], several grid assembly packages have become available. They include dedicated assembly codes such as PEGASUS5 [24], SUGGAR++/DiRTlib [25], CHIMPS [26], and PUNDIT [27]. Other

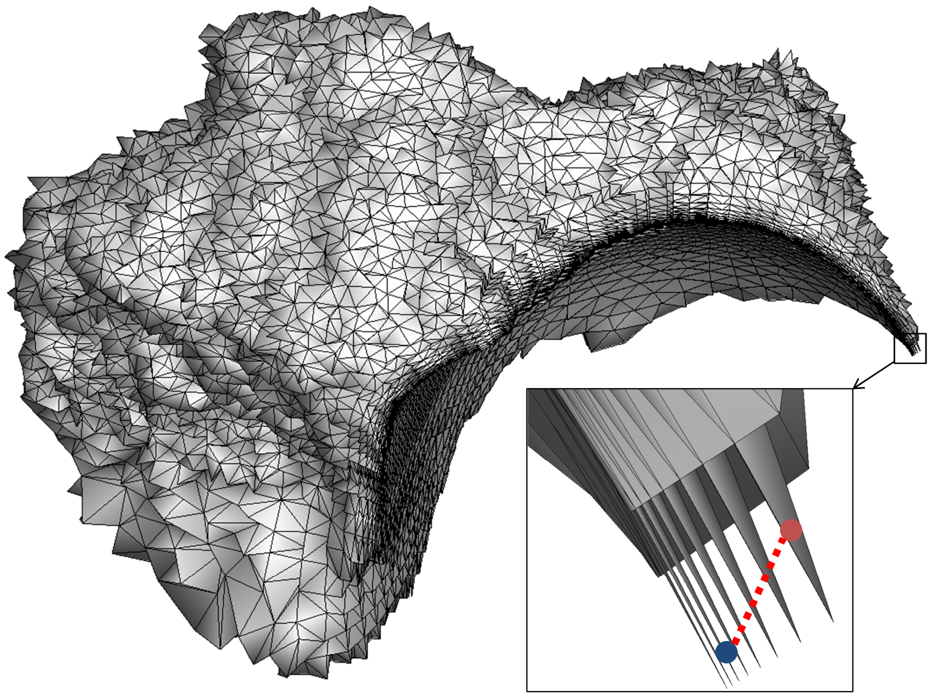


Figure 1.1: Example of unstructured mesh-block partition illustrating a case of multiple exit/re-entry points during line-walk based donor search.

assembly codes have been directly integrated within a specific solver, such as in OVERFLOW [29], OVERFLOW-D [28], BEGGAR [30], FASTRAN [31], Overture [32], and ElsA [33]. All these codes have their own specific merits and limitations. One prominent limitation is lack of modularity: many of these codes are intended to be used within a single solver and cannot handle multiple solvers and multiple mesh types simultaneously. Another common limitation is the inability to support mixed mesh types (unstructured and structured). In addition, some codes are not implemented in parallel and hence can not practically handle large scale problems. Further, often a different paradigm for parallelism, such as maintaining the entire meshes in each process (coarse grain parallelism) or using a different partitioning scheme (compared to solver based partitioning) for overset grid assembly is utilized to simplify the overset grid assembly process. However, this would require replicating and remapping the entire grid data causing both memory and execution efficiencies that can become especially prominent in unsteady flow computations. Another limitation is the lack of full automation, particularly for the task of removing invalid points inside solid walls (termed hole-cutting), for which most codes require some amount of user input and expertise.

The present paper describes an overset grid assembly method that attempts to overcome the above challenges, by developing efficient and robust algorithms to perform each sub-task of the global parallel Overset Grid Assembly problem in a fully automated manner, requiring no user input other than the partitioned mesh system itself, distinct body tags associating them to different bodies and locations on the mesh system where the flow boundary conditions are to be applied. It is to be emphasized that only local information, i.e. on a per process basis, is sought for non-Cartesian meshes, with no requirement for any global mesh data of any kind. This method is currently implemented in PUNDIT [27], which is the grid connectivity module of Helios [34], the rotary-wing product of the CREATE-AV [35] and the HPC Institute for Advanced Rotorcraft

Modeling and Simulation (HI-ARMS) programs.

The paper is organized as follows: in the first chapter, the problem of Overset Grid Assembly on a partitioned grid system is described, and the terminology used in the paper is defined. In the second chapter, the proposed method is described, which consists of five steps: hole profiling, mesh-block profiling, donor search, point type assignment, and finally interpolation. The problem of parallel overset grid assembly is inherently imbalanced because of large differences in the amount of work between different processes. The major contribution of this work is in developing a active load-balancing capability that can mitigate this imbalance and improve the scalability and robustness of the proposed method. The third chapter presents results of the load-reblance algorithm that we have developed. Traditional overset grid methods are known to be non-conservative because of problems with interpolation. We have recently developed and tested (in 2-D) conservative overset method for cell-centered unstructured grid problems. The final chapter of this document delves on this subject.

Chapter 2

Problem Definition

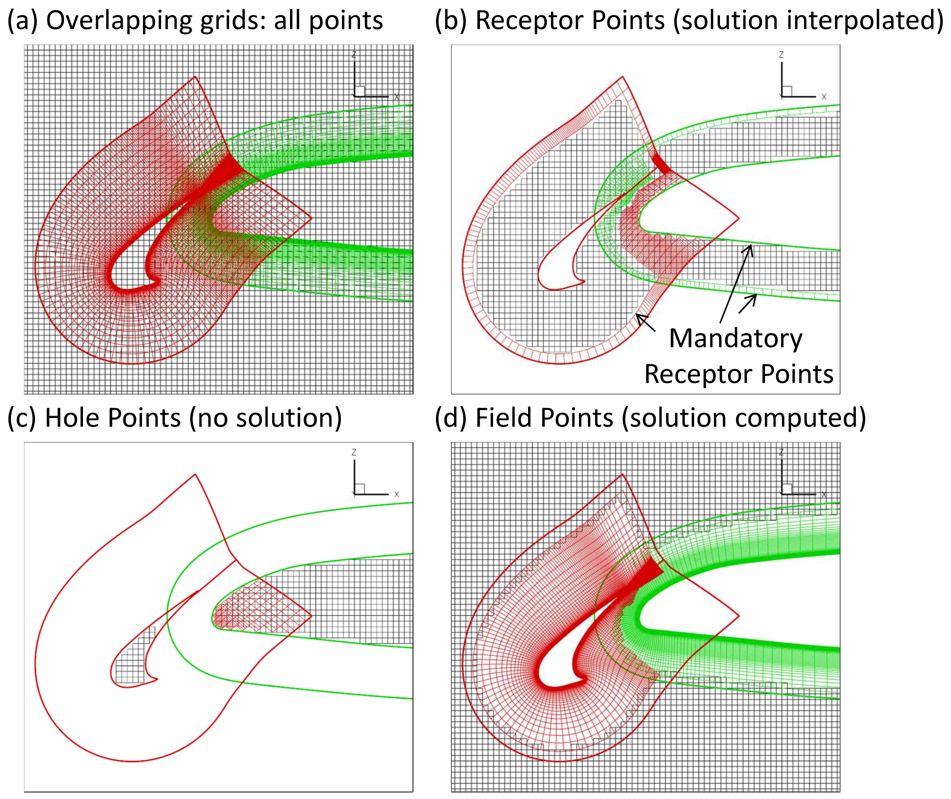

(a) Overlapping grids: all points
(b) Receptor Points (solution interpolated)
Mandatory
Receptor Points
(c) Hole Points (no solution)
(d) Field Points (solution computed)

Figure 2.1: Example of point types: Receptor points, Hole Points, and Field Points

Consider a mesh system composed of several types of meshes, e.g. unstructured, structured and Cartesian meshes, that overlap each other. The entire mesh system is partitioned and distributed such that each process, in general, owns few mesh-blocks (that could be of any or all types). The Overset Grid Assembly (OGA) problem consists of processing this partitioned mesh system in a parallel computing environment such that dominant meshes are identified in regions of overlap. The final outcome is the identification of mutually exclusive regions in each mesh block, where, (a) the flow field needs to be computed (b) the flow field needs to be interpolated from another grid and (c) the flow field is invalid. Specifically, the primary task of the overset grid assembly module consists of assigning to each mesh node (or mesh cell, if the solution variables are located at the cell center), one of three point types, with the following properties:

1. *Hole point* (also termed blanked-out point): point at which no solution is computed.
2. *Field point* (also termed active point, or solver point): point at which the governing partial differential equations are discretized and solved.
3. *Receptor point* (also termed interpolation point, or fringe point): point at which the solution is interpolated from an overlapping cell (termed donor cell).

In order to establish the type of each grid point in an entirely automated manner, requiring no user input other than the grid characteristics, the following definitions are used:

- **Query Point**:
  point used in the flow solver to compute the flowfield (can be a node or cell-center depending on the location of flow variables), and for which a point type is sought.
- **Candidate donor cell** (for query point P):
  any cell (of any mesh other than the mesh P belongs to) which contains P.
- **Donor cell** (for query point P):
  the cell with finest resolution (nominally quantified by the volume of the cell) among all candidate donor cells for P.
- **Hole Point**:
  query point located inside a solid body.
- **Field Point**:
  query point which is not a hole point and either does not have a donor cell or has a donor cell of coarser resolution than itself.
- **Mandatory receptor point**:
  query point which is located within a certain number of cells (based on the breadth of the stencil used in discretizing the governing equations) of either:
  - an outer grid boundary node, or
  - a hole point.
- **Receptor point**:
  query point which is not a hole point and either is a mandatory receptor point or has a donor cell of finer resolution than itself.
- **Orphan point**:
  mandatory receptor point which does not find a donor cell (undesirable, the overset grid assembler should create no such point).

The three different point types are illustrated in Figure 2.1, which shows two overlapping structured meshes (near-body since they conform to a no-slip wall boundary) and a Cartesian background mesh (off-body since they are not body conforming) before and after domain connectivity processing. Using the above set of definitions, field points are automatically selected such that there is minimal overlap between the different grids (Figure 2.1(d)).

Chapter 3

Load rebalance algorithm

3.1 Problem definition

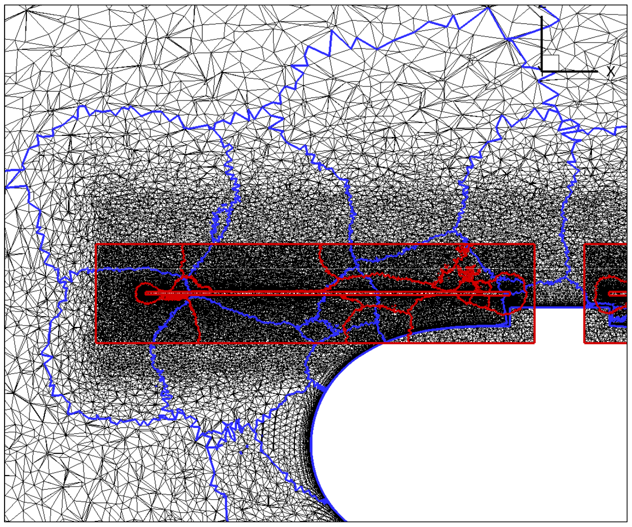


Figure 3.1: Hart-II mesh-blocks partitioning: blade detail.

For highly non-homogeneous meshes, the query points distribution among the mesh-blocks can be very imbalanced. This is partly because the grid partitioning is typically done by maintaining a constant number of cells per partition, leading to a large variation in mesh-block volume. A large mesh-block overlapping several small mesh-blocks can get an undesirably large number of query points, while a mesh-block without any overlap (or only background Cartesian mesh overlap) will not get any query point. For example, Figure 3.1 shows a detail of the HART-II [41] region where fuselage blade mesh-blocks overlap. It is clear that some fuselage mesh-blocks (with large blade mesh-block overlap) will be overloaded, while others will have no load at all. The resulting load imbalance can be alleviated using data exchange between the processors. A load re-balance algorithm is designed, which operates as follows.

The first step is to estimate the load per processor. This load is measured as the total time required to perform the connectivity task. When several mesh-blocks are assigned to a processor, the loads per mesh-block are also measured. The average load among all processors, L_A is then computed.

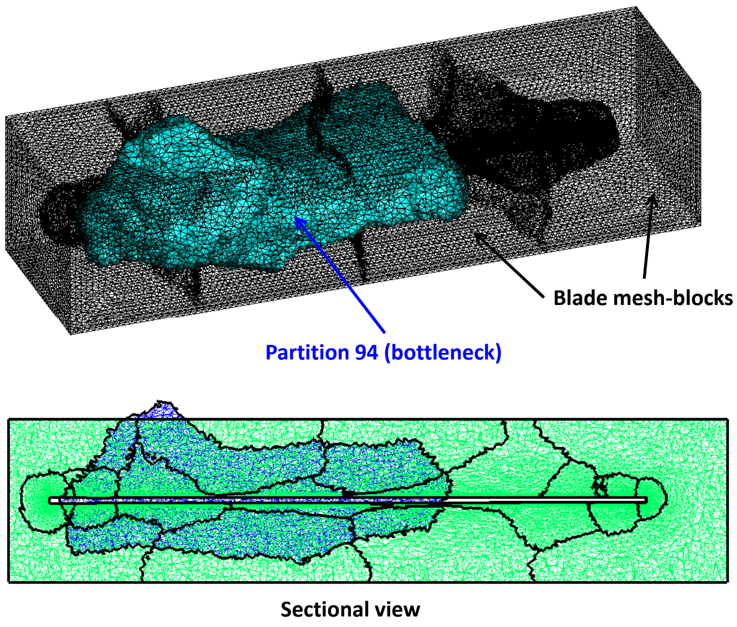


Figure 3.2: HART-II: bottleneck partition before load re-balance.

The next step is to perform the initial load re-balance. Exchange tables describing which processor should transfer which fraction of its load to which processor(s) are determined in the following iterative way: The most loaded processor (P_d, with load L_{max}) transfers a load ΔL to the least loaded processor (P_r, with load L_{min}), until one of them reaches the average load: $\Delta L = min(L_{max} - L_A, L_A - L_{min})$. The loads on both donor and receptor processors are then updated accordingly and this is repeated until the load on each processor is within a small fraction of the average (10% in the present study).

The resulting exchange tables describe the desired data transfer between processors. Actual data transfer is between a mesh-block and a processor, so the tables are then transformed to establish the exchange tables between mesh-blocks and processors, based on the stored mesh-block loads: i.e. for each mesh-block, which fraction needs to be transferred to which processor. The reverse list is also easily established: for each processor, which mesh-block(s) are sending data.

In order to partition the overloaded mesh-blocks into "sub-mesh-blocks" to be sent across to less loaded processors, we assume that the load is directly proportional to the number of query points. In other words, if a mesh-block must transfer x% of its load, it sends across x% of its query points (along with the cells overlapping them). Note that this assumption can be inaccurate, because some query points carry a smaller computational cost than others (for example, those having no true overlap with cells, i.e. inside empty auxiliary grid sub-blocks). However, it is reasonable enough to yield good initial re-balance results at the first iteration. In subsequent iterations, an adaptive load-rebalance algorithm is applied to further correct the remaining load imbalance.

In order to efficiently determine which query points and cells to send across, each donor mesh-block is pre-processed using a special type of auxiliary grid. This AG, similar to the one used during the EIM donor search, has outer bounds corresponding to the oriented bounding box of the query points, but it is only divided in the direction with the largest extent. The number of sub-blocks is chosen equal to the number of query points. The list of query points per sub-block is easily determined, then used to compute the levels at which the mesh-block should be partitioned

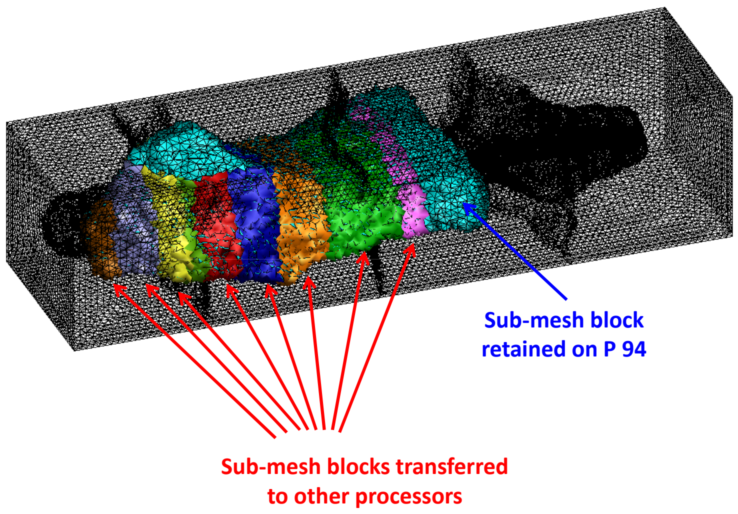

Sub-mesh block retained on P 94
Sub-mesh blocks transferred to other processors

Figure 3.3: HART-II: bottleneck partition after load re-balance.

to closely match the load specified in the data exchange tables. Once the sub-mesh-blocks are identified, the list of cells per sub-mesh-block is determined. Finally, the data to be sent across (query points and cells) is packaged into two arrays of real and integer numbers. Note that while the ADT donor search method only requires query point and cell information, other methods such as EIM require additional data such as cell neighbors, etc. For this reason, in the present work, the load-rebalance algorithm is limited to the ADT method, and will later be extended to more efficient donor search algorithms.

Processor-to-processor communication is performed using Message-Passing Interface routines. In order to increase efficiency, only non-blocking communication is used. Additionally, donor processors are made to check periodically for completion of their send requests in order to allow the receptor processors to initiate their received task as early as possible. In this way, a donor processor posts its send request before immediately (while the data is being sent) starting to process its remaining mesh-blocks (or fraction of mesh-blocks). A receptor processor first processes its own mesh-blocks, then posts a receive request. As soon as the associated send request is completed, it processes the received data and sends back the donor information, which is then received by the donor processor and used to update its final list of donor cells.

The mesh-block re-partitioning process is illustrated using the HART-II [41] rotor-fuselage case with 256 processors. Figure 3.2 shows two views of the bottleneck processor (P 94). It consists of a single mesh-block, almost fully immersed inside the bounding box of all blade 1 mesh-blocks (potential overlap with 19 mesh-blocks). After re-balance, as shown in Figure 3.3, it has been sliced along the longest extent of its bounding box (along the blade) into 8 sub-mesh-blocks, each sent to a different processor. Note that a mesh-block portion still remains assigned to processor 94.

After the initial load re-balance is complete, results can still be inadequate. This is in part because, as mentioned earlier, not all query points carry the same computational cost. In addition, new tasks have been introduced (determining new sub-partitioning, communication tasks), which were not accounted for in the initial load measurements. In order to further improve load balance, an adaptive algorithm is applied for the sub-sequent iterations. This algorithm uses the current load measurements to correct the previous data exchange matrix, in the following way:

- The new load average is first computed (based on current measurements, i.e. reflecting previous load-rebalance operations)

- The most loaded processor P_max identifies the load (ΔL) to transfer to the least loaded processor P_{min} so that one of them reaches the average.
  - if P_{max} was previously a donor processor (or inactive), and P_{min} was a receptor processor (or inactive), P_{max} simply increases the load transferred to P_{min} by ΔL.
  - if P_{max} was previously a donor processor (or inactive), but P_{min} was also a donor processor, two corrections are required:
    - P_{max} transfers the load to one or several of the receptors of P_{min}
    - P_{min} transfers a correspondingly reduced load to these receptors
  - if P_{min} was previously a receptor processor (or inactive), but P_{max} was also a receptor processor, two corrections are required:
    - One or several of the donors of P_{max} transfers the load to P_{min}
    - P_{max} receives a correspondingly reduced load from these donors
  - if both P_{min} and P_{max} were of different types at the previous iteration (i.e. P_{max} was a receptor and P_{min} was a donor), three corrections are required:
    - One or several of the donors of P_{max} transfer the load to one or several receptors of P_{min}
    - P_{max} receives a correspondingly reduced load from these donors
    - P_{min} transfers a correspondingly reduced load to these receptors
- This is repeated until all corrected loads are within a fraction of the average load.

This adaptive algorithm also allows the load-rebalance to remain applicable to moving/deforming grid problems.

3.2 Results

Results of the Overset Grid Assembly task, in terms of total time, robustness, and scalability, are obtained for two realistic application cases: the HART-II rotor-fuselage configuration [41], and a generic wing-pylon-store configuration [42].

3.3 Timing and Robustness Results

The HART-II case, illustrated in Figure 3.4, consists of 5 near-body unstructured meshes (4 blades and one fuselage). There is no off-body mesh, instead the fuselage mesh extends a large distance away from the fuselage. The total number of nodes in all near-body meshes is 7 millions. Figure 3.4 also presents a detail of the meshes before and after connectivity (showing only the solver cells), using the EIM method. It can be observed that only the most suitable cells from each mesh are chosen for computation, automatically resulting in minimum overlap between the processed meshes.

This case is run on 256 processors, and the timing results are presented in Figure 3.5. In this figure, timings corresponding to the main Overset Grid Assembly tasks are represented for each processor in stacked bar plot format. The first task is hole profiling (HP). The second task correspond to mesh-block profiling (MBP) and the third task is donor search (SRCH). The fourth task corresponds to communication performed to identify the correct point types (Comm), and

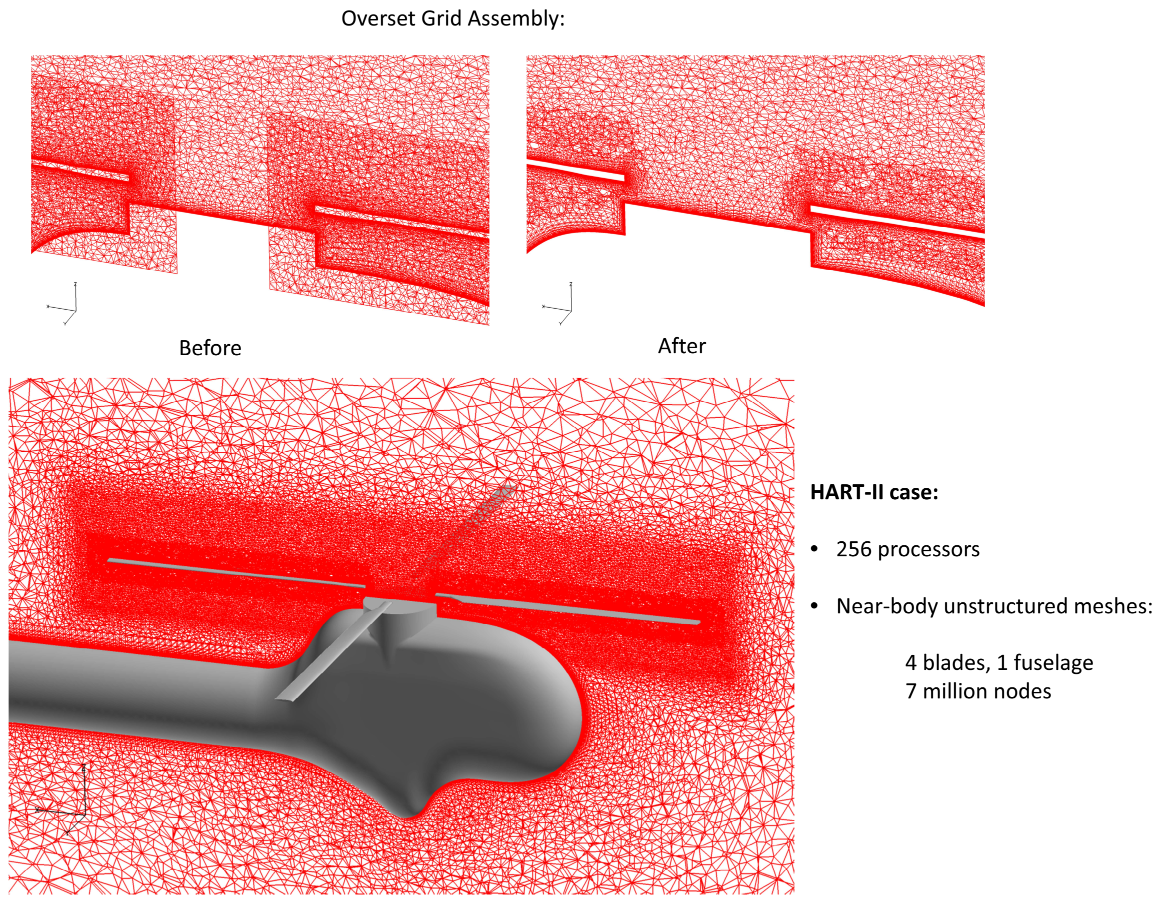


Figure 3.4: HART-II rotor-fuselage case

finally the last task consists of computing the interpolation weight for all points identified as receptor points (Interp).

Note that the hole profiling task accounts for a negligible fraction of the total OGA time. A significant load imbalance is observed using both methods. The EIM is more efficient than the ADT method, with a OGA time of 1.3 seconds (7% of the total time step time), compared to 2.6 seconds for the ADT method (14% of the total time). The connectivity results are verified to be identical for both methods (same number of points of each type).

3.4 Load re-balance and scalability Results

Figure 3.6 shows results of the load rebalance algorithm when applied to the HART-II case. Task durations are shown for the first three iterations (before load re-balance, after initial load re-balance, and after first adaptive load re-balance). For clarity, only the mesh-block profiling and donor search tasks are shown for iteration 1. For iterations 2 and 3, additional load re-balance tasks are included:

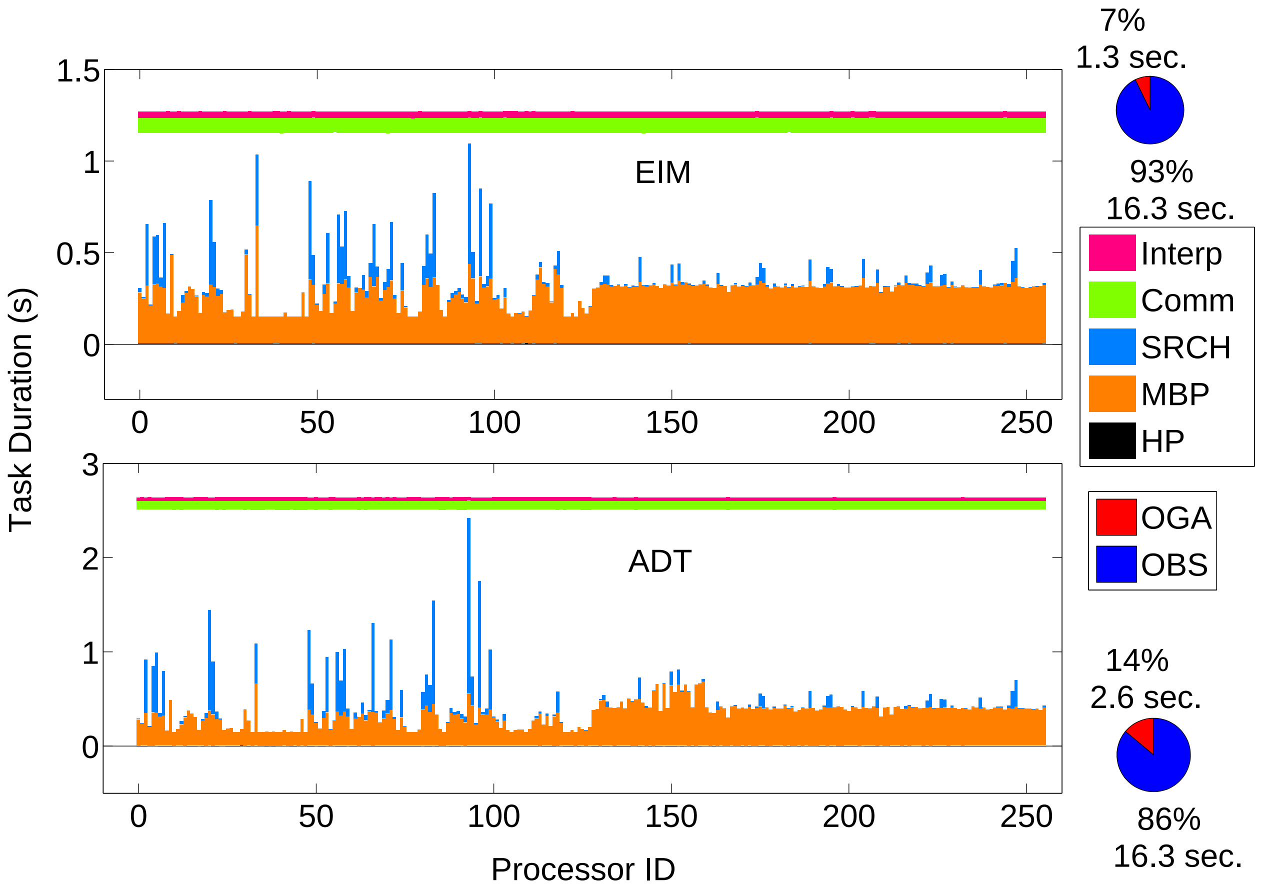


Figure 3.5: Timing results for the HART-II case using two different methods. Bar plots represent the timing per processor for each OGA task and pie charts show the total time per iteration, split into OGA, OBS (off-body solver) and NBS (near-body solver).

identification of data to transfer and donor update (over-loaded processors), and processing of the received data (under-loaded processors). The task of computing the data exchange matrix (after all processors complete their connectivity tasks) is also pictured (negligible time). After the initial load-rebalance, the total time decreases from 2.1 sec to 0.7 sec (68% reduction), and futher decreases to 0.5 sec after the third iteration (76% reduction). Figure 3.6 illustrates the efficiency of using non-blocking communication: donor processors can process their own data immediately after identifying the data to be transferred, and receptor processors process their own data before waiting a minimal time to receive additional data. Figure 3.7 shows the variation of total time required to perform the grid assembly before and after load re-balance, for different numbers of processors (32, 64 and 256). Some improvement in scalability is demonstrated using load re-balance, with a speed-up of 160 (instead of only 73) using 256 processors.

Finally, the scalability of the method is also tested using a Wind-Pylon-Store configuration illustrated in Figure 3.8. This test case consists of 3 unstructured grids around a wing and two stores, with a total of 15 million cells. Figure 3.8 shows the grids before and after connectivity, including a detail around the wing-pylon-store region of overlap. The overset grid assembly task is performed using 64, 128, 256 and 512 processors and timing results are shown in Figure 3.9. After load re-balance, the total time is reduced by up to 80% (for 256 processors). The speed-up is also

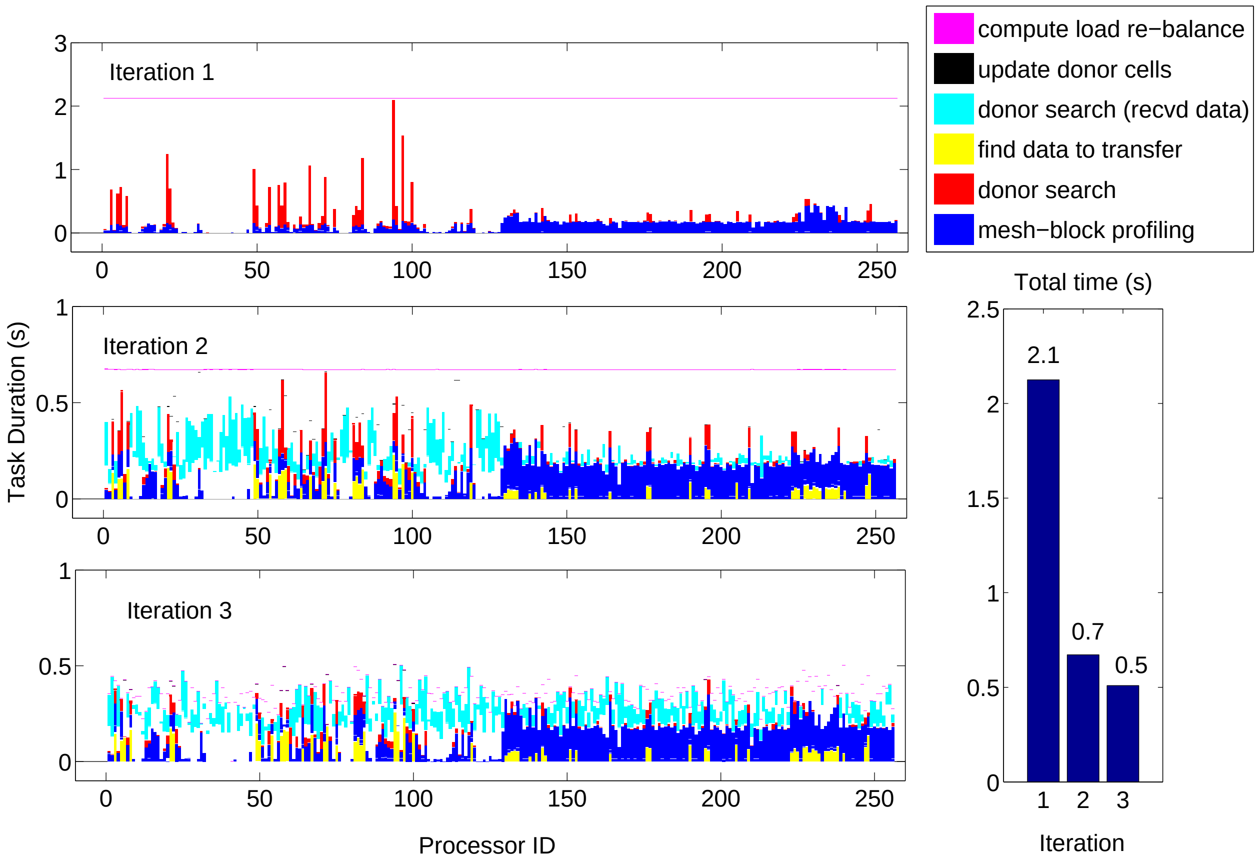


Figure 3.6: Task durations before load rebalance (iteration 1); after initial load rebalance (iteration 2); and after first adaptive load re-balance (iteration 3).

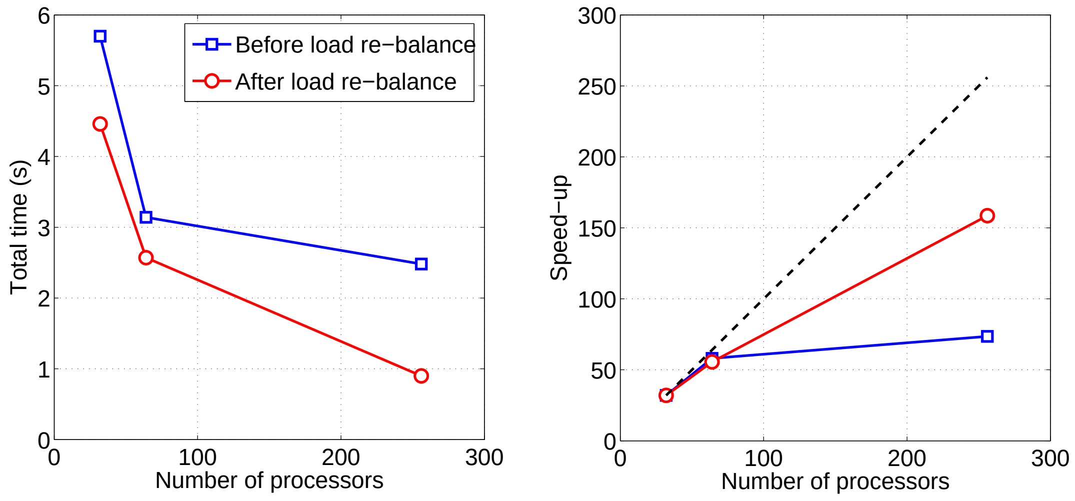


Figure 3.7: HART-II case: time to perform OGA task before and after load re-balance.

improved, with a value of 213 using 256 processors (instead of 117 without load re-balance), however the scalability remains significantly below the ideal linear trend, indicating further improvements to the algorithm are required.

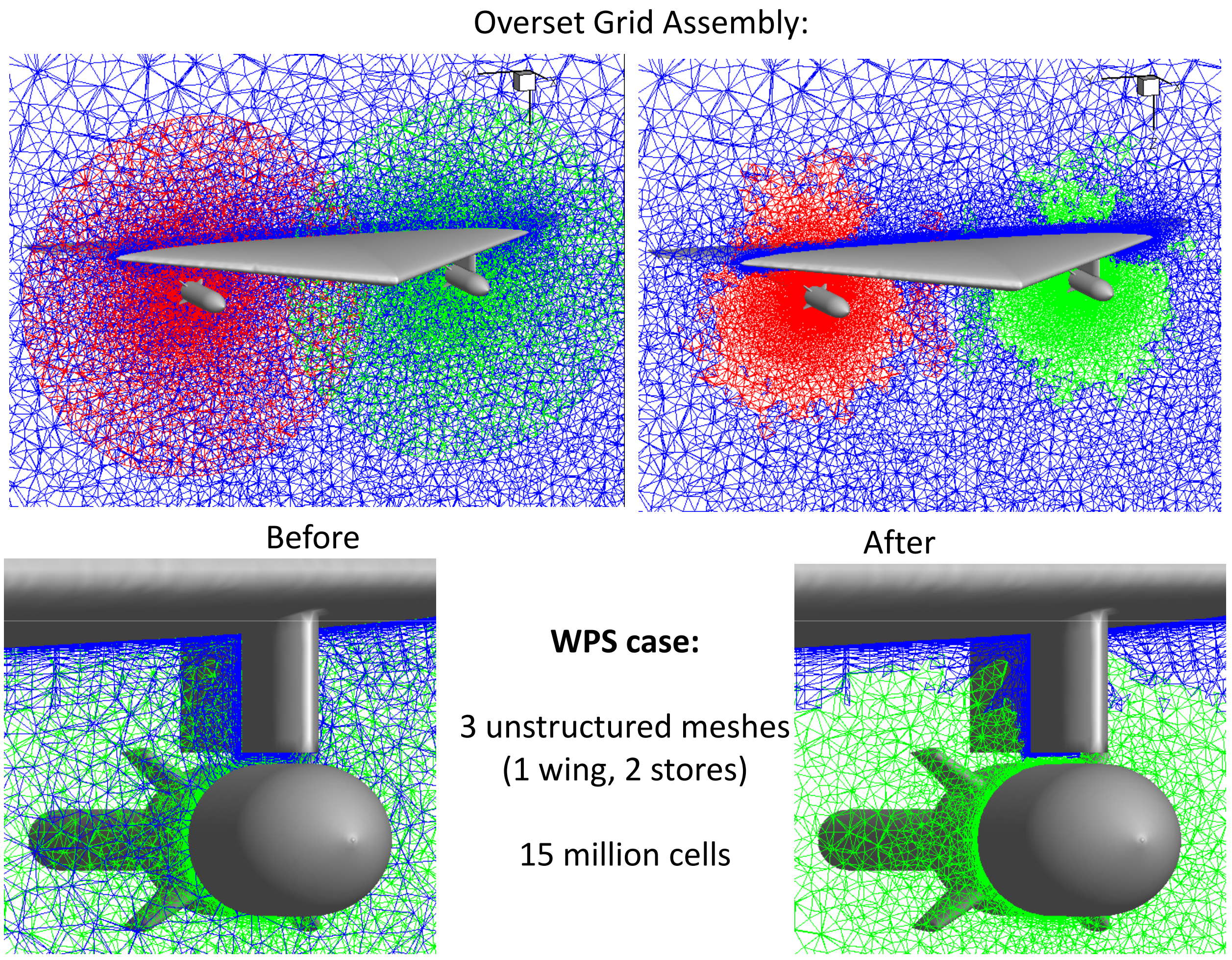


Figure 3.8: Wing-Pylon-Store case: connectivity results

3.5 Conclusions

This chapter presents a method to perform Overset Grid Assembly on a system of overlapping unstructured meshes. These meshes are assumed to be partitioned into multiple mesh-blocks and processed on multiple cores. The proposed OGA method uses structured, homogeneous auxiliary grids to build exact inverse maps (EIM) that are used to speed up donor searches. This method is general (can handle several mesh types), robust (no undesirable point is generated), fully automated (only the meshes and solver information are required), and modular (can be integrated with multiple solver). The proposed EIM method is compared with the Alternating Digital Tree (ADT) method using the HART-II rotor-fuselage configuration as the main test case. A large load imbalance is observed, reflecting the very imbalanced distribution of query points in the case of highly non-homogeneous meshes. The proposed method is shown to be more efficient than the ADT method (by a factor of 2 for the HART-II case), without loss of robustness (identical number of point types identified). An adaptive load re-balance algorithm is described and tested, which demonstrates

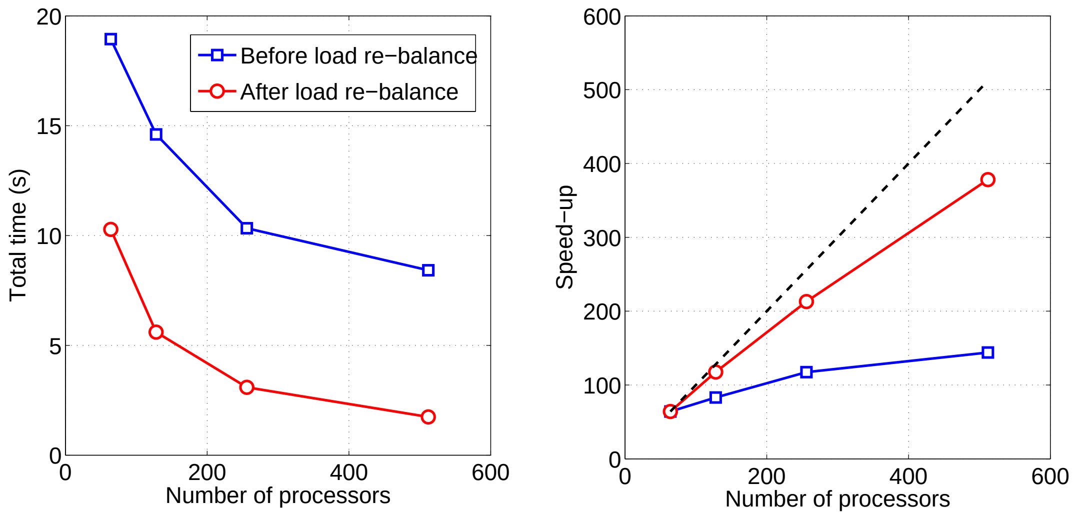


Figure 3.9: WPS case: time to perform OGA task before and after load re-balance.

improved efficiency (total time reduced by 76% for HART-II) and scalability (speed-up increased from 117 to 213 using 256 processors for a wing-pylon-store test case). The re-balance algorithm is at present implemented using the ADT method, and work is currently under way to extend it to the more efficient EIM method.

Chapter 4

Conservative overset grid methods

One of the problems of the overset mesh techniques is the lack of conservation. The reasons are the geometrical inconsistencies at the boundaries of the meshes to each other and the data transfer via interpo- lation of variables within meshes. This problem becomes worse when the discontinuities cross the intergrid boundaries. Furthermore, high order interpolation schemes causes numerical oscillations near discontinuities such as shocks due to high gradients. It is possible to interpolate uxes instead of conservative or primitive variables however, this approach is cumbersome and limited to certain problems. Instead of using inter- polation schemes, the meshes could be united by removing the intergrid elements and then remeshing to obtain a single mesh. By this way, the method will become conservative automatically. This method has been successfully applied on the node-centred unstructured meshes while in the present work cell-centred meshes will be used. The main dierence between the present work and8 is that for node-centred meshes after remeshing the number of variables remain the same while for the cell-centred meshes, at each time step, new variables are introduced. Identication of the active and non-active regions are performed with alternating digital trees which are also used for advancing front technique to remesh the overlapping grid region. Two test cases, oscillating single airfoil and airfoil- ap conguration in relative motion are used are used to validate the new method. Results are compared with the ones of traditional overset method.

4.1 Overset Mesh Technique

Field, fringe, and the hole cells are identied with respect to the distances from the boundaries to avoid cell islands. To speed up the process an alternating digital tree (adt) is used. For the conventional overset grids method, variables are interpolated between the grids with second order accuracy. This is the main problem of conventional method especially when a discontinuity such as a shock crosses the overlapping region.

4.2 Remeshing

After the identication of the active and non-active cells, some of the cells are removed from the grids leaving a blank region which is approximately twice the local mesh size. This blank region is remeshed with advancing front technique which is boundary conforming. Meshing by using only the existing points results in relatively greater volumes and possibly high aspect ratio cells, hence new points are created where appropriate during the meshing process. Being a cell-centered scheme after remeshing additional control volumes are created whereas in node-centered scheme,

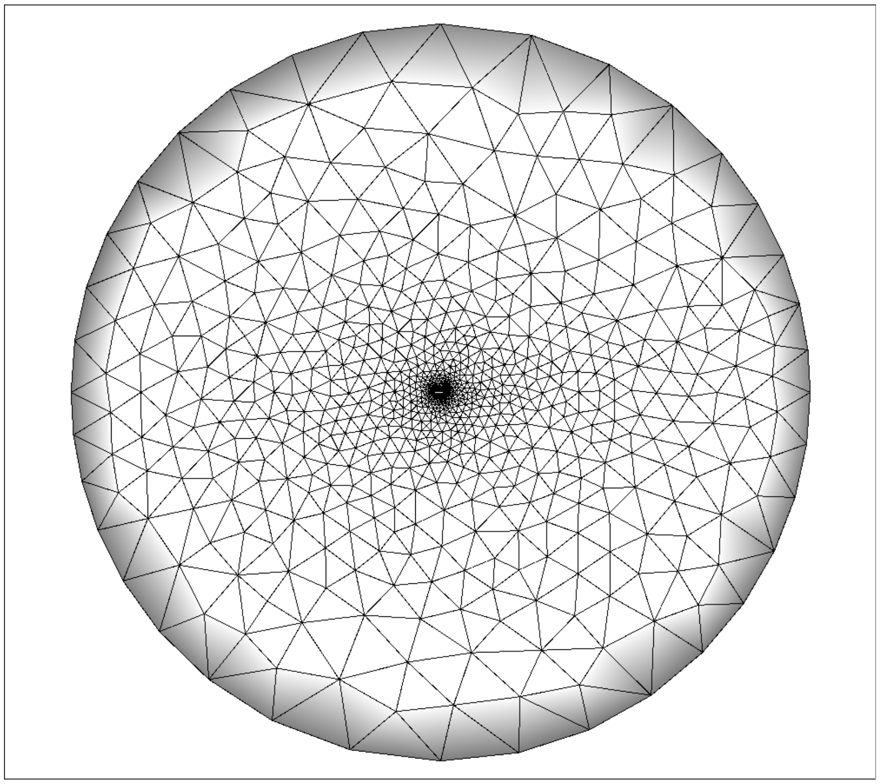


Figure 4.1: Airfoil grid

only the congurations of the control volumes change without creating additional points which can be considered as a downside of the cell-centered scheme. Having the blank region, each boundary point's and edge's belongings are assigned with the grid number. This procedure is also applied to the new points and edges. For example, some of the points and edges belong to grid 1 and some of them to grid 2 whereas the new elements will belong to grid 3. These points and edges are stored in a point and a front list which is sorted according to primarily the grid belongings and as a secondary condition according to length of edges. So, the boundary edges have the priority to meshed rst however, if two edges belong to the same grid then the smaller edge is meshed rst. Unless, there is no other point in the radius of a candidate new point, that new point will be created. The radius is the half of the average mesh size which is found by taking the average of all intergrid boundary grid edges. The new point is created at average mesh size in normal direction to the center of the edge. The direction of the normal is chosen so that the new triangle which is going to be formed will be inside the domain and does not intersect with other edges and triangles. Created edges and triangles are added to the front and triangle lists, respectively while the already used front list member is removed from the front list which is sorted consequently. This procedure is repeated until there is no edge lefts in the front list. If a new point cannot be created, the existing points are used by searching for a closest point to either one of the terminals of the front list edge. If a triangle can be formed with both of the closest points to the terminals then the closest point which results in a smaller triangle is chosen. Once the new grid formed, the connection between the new grid and the already existing grids are established.

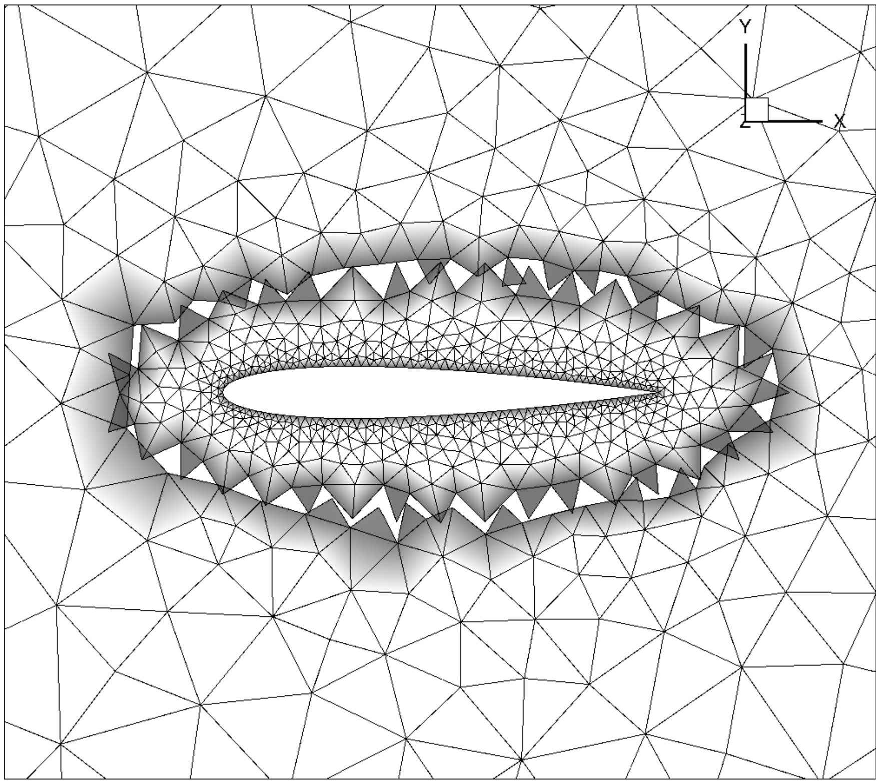


Figure 4.2: Overset grids

4.3 Unsteady Term of New Cells

For the time-derivative term, the variables of the newly created cells at previous time levels should be available. Since the rotation displacement is known, the corresponding cell at the previous time level is found by an alternating digital tree and the previous time-level variables are interpolated from that cell with second order accuracy using the gradients.

4.4 Validation

4.4.1 Transonic Moving Airfoil

To validate moving mesh algorithm and also see the consequences of using single grid, conventional overset grids, and remeshed grid, the mentioned methods are tested with the case of moving airfoil through stationary

uid at M = 0.8 and α= 1:25. The airfoil grid which is shown in Figure 1 has a radius of 30 chord length and consists of 2522 triangles while the side length of the square background mesh is 60 and has 2194 triangles. The overset grids after the identication of active and non-active cells are shown in Figure 2. For the new method, the remeshed grid is shown in Figure 3. The overlapping region is located where there is a shock present. This is done intentionally to show the superiority of the remeshed grid over overset grids. The pressure coe cient variations on the airfoil are shown in Figure 4 in which the single grid and the new method values are almost identical while the conventional method results are clearly different. The sub-iteration convergences are presented in

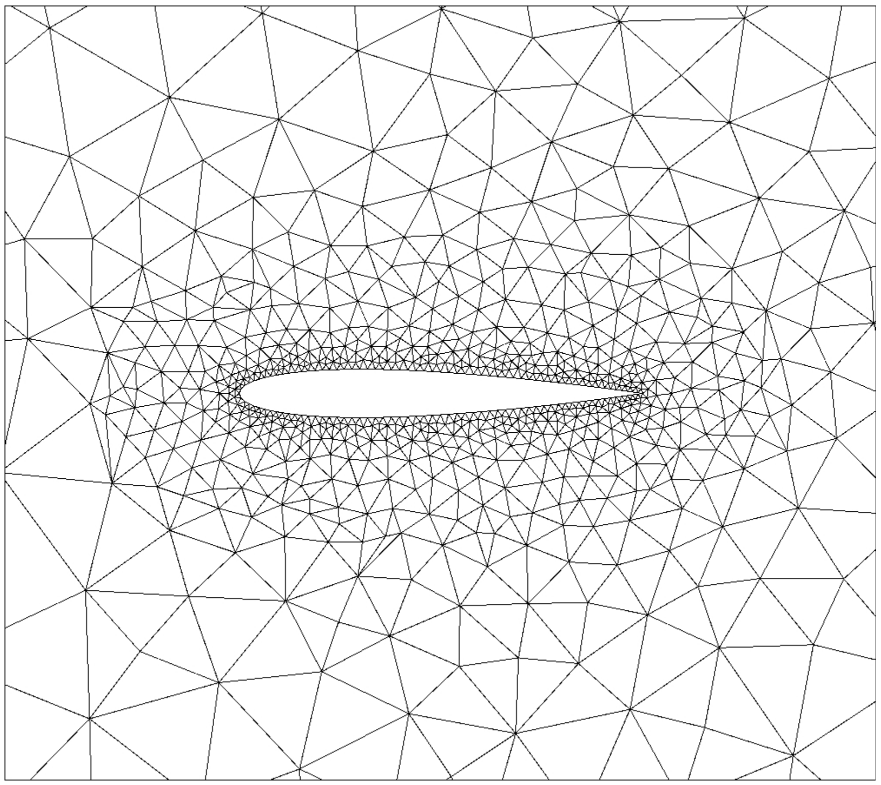


Figure 4.3: The grid obtained after filling the blank region

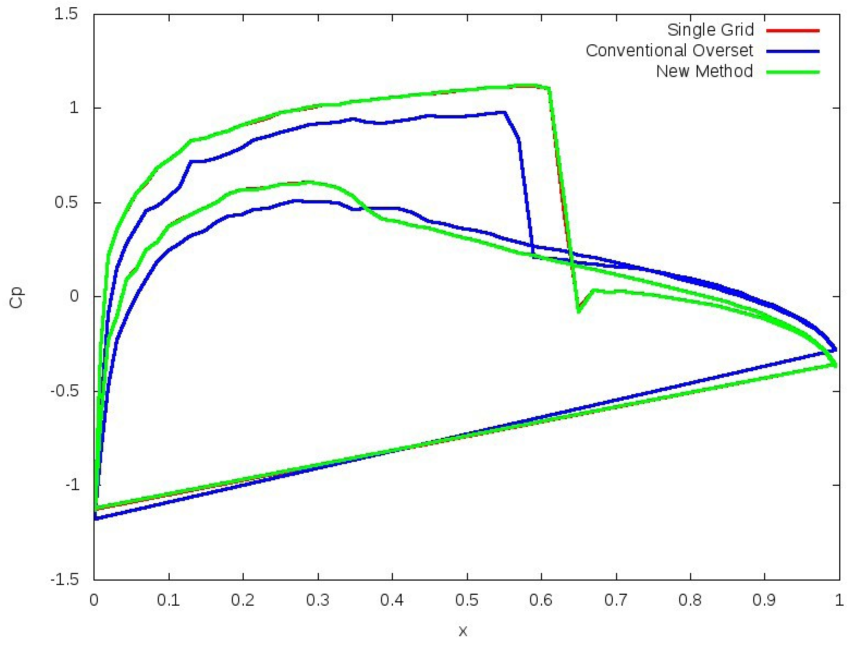


Figure 4.4: Pressure coefficient distribution for transonic moving airfoil

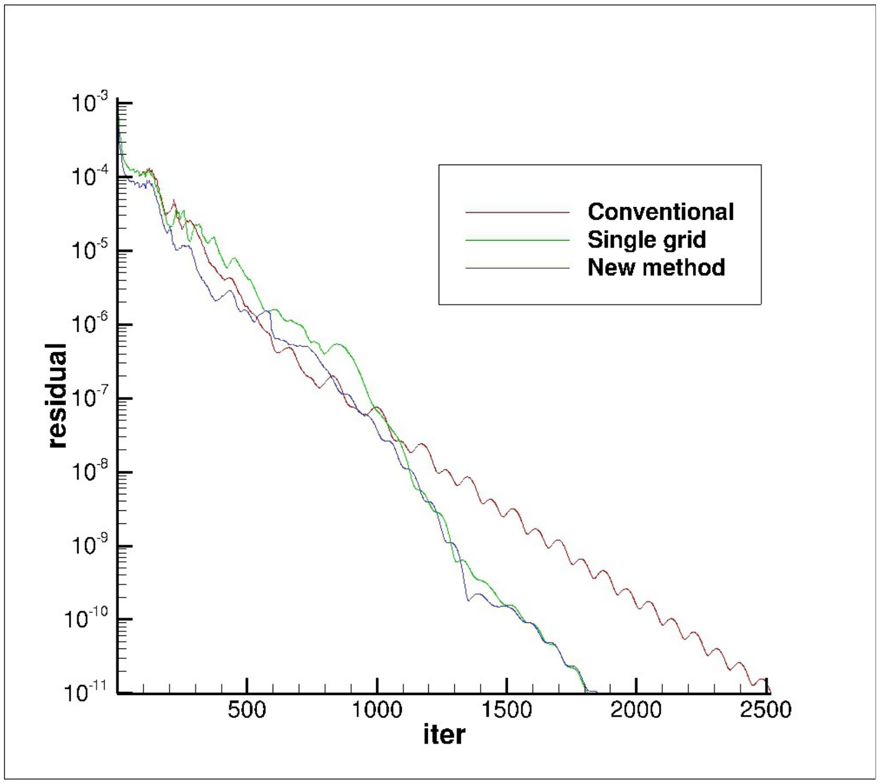


Figure 4.5: Convergence of the flow solution for transonic moving airfoil

Figure 5. It is clear that the conventional method converges slower than the single grid method and the new method.

4.4.2 Oscillating Airfoil

In addition to the first test case, together with translational motion, the airfoil mesh is also rotating. The angle of attack of the airfoil is defined as:

$$\alpha = \alpha_0 + \alpha_m(2k_cMt)$$

where $k_c = 0.0814$ is the reduced frequency, M=0.755, mean angle of attack α_0=0.016, and amplitude is α_m=2.51. Figures 8 to 13 show the density contours at various angle of attacks for all new method. Lift coefficient vs angle of attack is shown in Figure 6, where the conservative method can be observed to show good agreement with the single grid approach compared to the conventional overset grid method. Figure 7 shows the density contours at different time levels of oscillation for the three methods. Once again, it is evident that the flow field predicted by the conservative method agrees better with the single grid method when compared with the traditional overset method.

4.5 Concluding Observations

Preliminary investigation of the conservative overset grid methods in a cell centered context show improved results when compared to conventional overset method. Work is expected to continue in

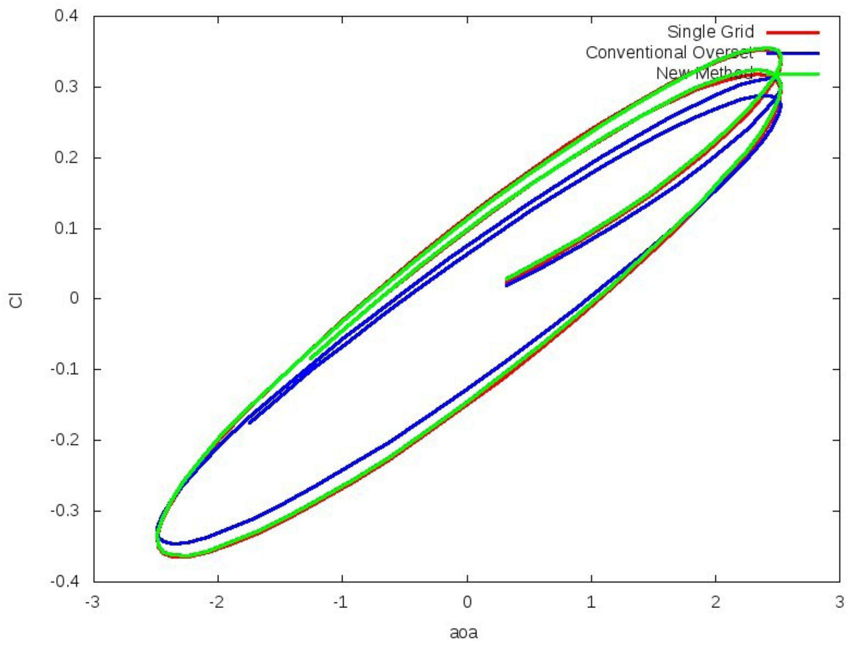
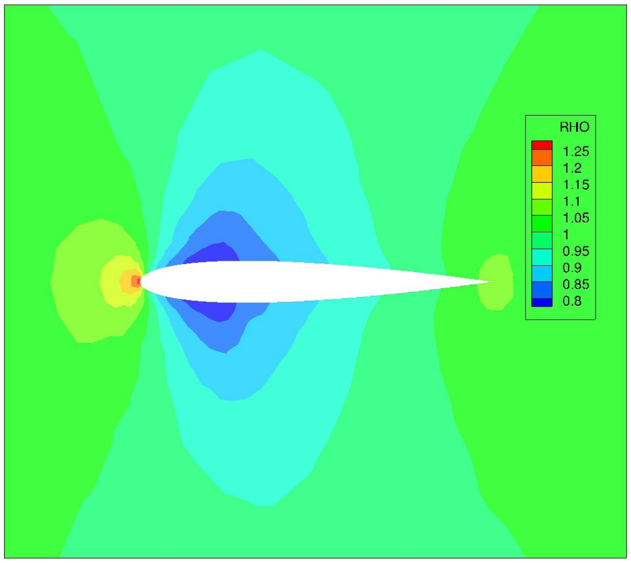
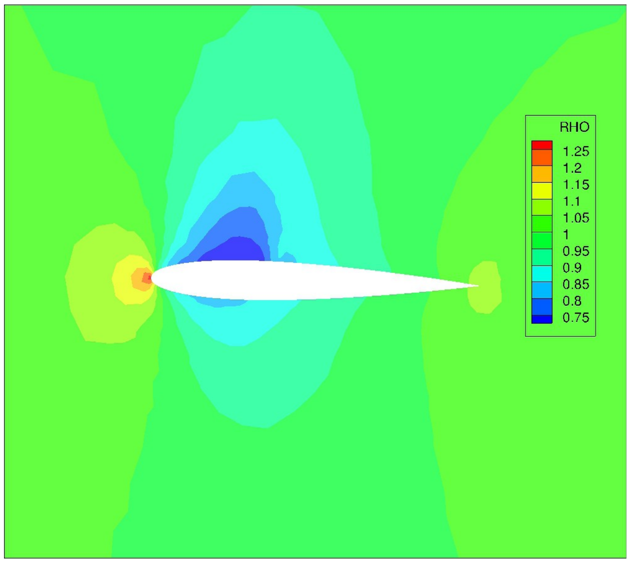


Figure 4.6: Lift coefficient w.r.t angle of attack for single mesh, conventional overset and conservative overset grid methods

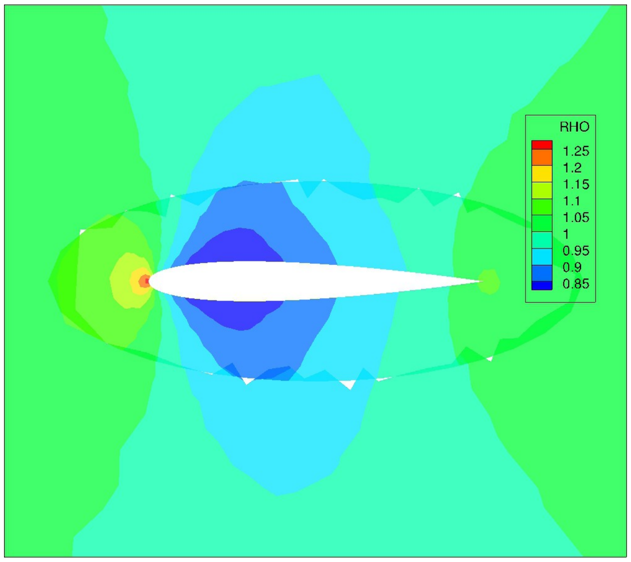
this front to further develop the techniques and improve the quality of the results.



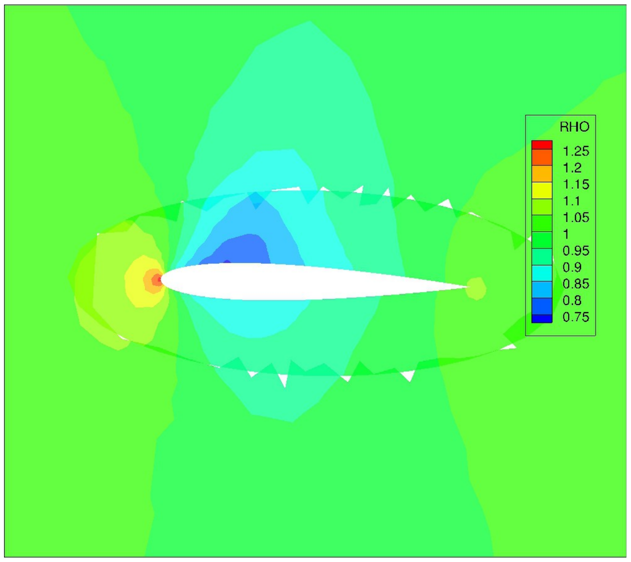
(a) α=0.32, single grid



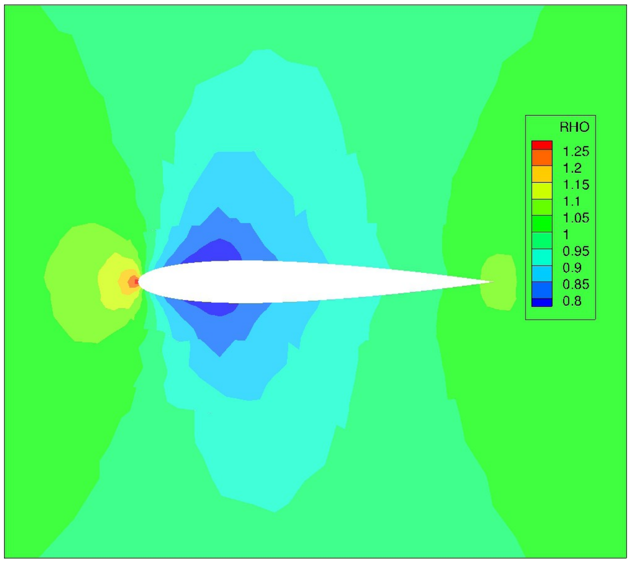
(b) α=1.70, single grid



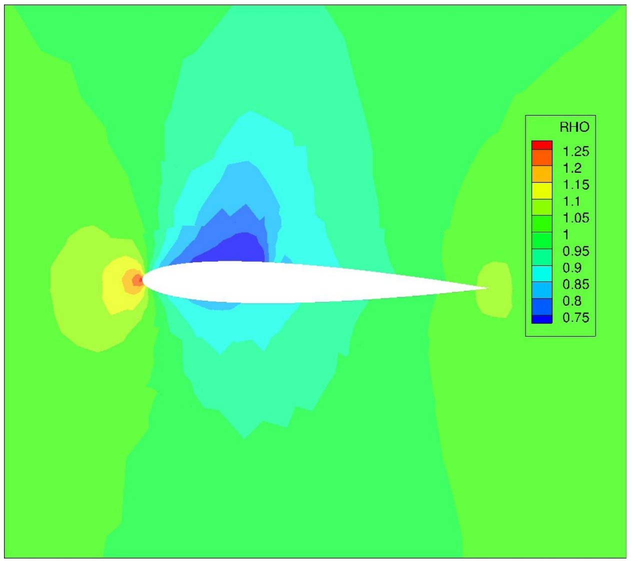
(c) α=0.32, conventional overset



(d) α=1.70, conventional overset



(e) α=0.32, conservative overset



(f) α=1.70, conservative overset

Figure 4.7: Comparison of density contours obtained for the oscillating airfoil problem

Bibliography

[1] Steger, J.L., and Benek, J.A., “On the use of composite grid schemes in computational aerodynamics”, Computer Methods in Applied Mechanics and Engineering, Vol.64, No. 1-3, October 1987, pp.301-320.

[2] Suhs, N., Rogers, S., Dietz, W.,“Pegasus 5: An automated pre-processor for overset-grid CFD”, American Institute of Aeronautics and Astronautics 41(6), 10371045 (2003).

[3] Noack, R.W., Boger, D.A., Kunz, R.F., Carrica, P.M., “Suggar++: An improved general overset grid assembly capability”, Proceedings of the 47th AIAA Aerospace Science and Exhibit, Orlando, FL (January 2009).

[4] Alonso, J. J., Hahn, S., Ham, F., Herrmann, M., Iaccarino, G., Kalitzin, G., LeGresley, P., Mattsson, K., Medic, G., Moin, P., Pitsch, H., Schluter, J., Svard, M., Van der Weide, E., You, D., and Wu, X., “CHIMPS: A High-Performance Scalable Module for Multi-Physics Simulations”, 42nd AIAA/ASME/SAE/ASEE Joint Propulsion Conference, AIAA, Washington, DC, July 2006.

[5] Sitaraman, J., Floros, M., Wissink, A., Potsdam, M. ”Parallel Domain Connectivity Algorithm For Unsteady Flow Computations Using Overlapping And Adaptive Grids,” Journal of Computational Physics, Volume 229, Issue 12, p. 4703-4723.

[6] Meakin, R.L., “Object X-Rays for Cutting Holes in Composite Overset Structured Grids,” 15th AIAA Computational Fluid Dynamics Conference, AIAA, Washington, DC, June 2001.

[7] Buning, P.G. and Pulliam, T.H., “Cartesian Off-body Adaptation For Viscous Time-Accurate Flow Simulations”, AIAA Paper 2011-3693, 20th AIAA Computational Fluid Dynamics Conference, June 27-30, 2011, Honolulu, Hawaii.

[8] Belk, D. M. and Maple, R. C., “Automated Assembly of Structured Grids for Moving Body Problems”, 12th AIAA Computational Fluid Dynamics Conference, Part 1, AIAA, Washington, DC, June 1995, pp. 381390.

[9] Wang, Z. J., Parthasarathy, V., and Hariharan, N., “A Fully Automated Chimera Methodology for Multiple Moving Body Problems”, 36th AIAA Aerospace Sciences Meeting, AIAA, Washington, DC, January 1998.

[10] David L. Brown, W. D. H. and Quinlan, D. J., “Overture: Object-Oriented Tools for Overset Grid Applications”, 17th AIAA Conference on Applied Aerodynamics, AIAA, Washington, DC, June 1999.

[11] JeanFaivre, G., Juvigny, X., and Benoit, C., “Parallel Chimera Computations of Helicopter Flows”, 24th International Congress of the Aeronautical Sciences, ICAS 2004.

[12] Sankaran, V., Sitaraman, J., Wissink, A., Datta, A., Jayaraman, B., Potsdam, M., Mavriplis, D., Yang, Z., O'Brien, D., Saberi, H., Cheng, R., Hariharan, N., and Strawn, R., "Application of the Helios computational platform to rotorcraft Flowfields", 48th AIAA Aerospace Sciences Meeting Including the New Horizons forum and Aerospace Exposition, 4-7 January 2010, Orlando, Florida.

[13] Post, D.E., "A new Dom initiative: the Computational Research and Engineering Acquisition Tools and Environments (CREATE) program", Journal of Physics, Conference Series 125, 2008.

[14] Zagaris, G., Campbell, M.T., Bodony, D.J., Shaffer, E. and Brandyberry, M.D., "A Toolkit for Parallel Overset Grid Assembly Targeting Large-Scale Moving Body Aerodynamic Simulation", Sandia National Laboratories, Proceedings of the 19th International Meshing Roundtable, 2010, pp.385-401.

[15] Pissanetzky, S. and Basombrio, F.G., "Efficient Calculation of Numerical Values of a Polyhedral Function", International Journal of Numerical Methods in Engineering, Vol.17, 1981, pp.231-237.

[16] Khoshniat, M., Stuhne, G.R., and Steinman, D.A., "Relative Performance of Geometric Search Algorithms for Interpolating Unstructured Mesh Data", MICCAI 2003, 6th International Conference on Medical Image Computing and Computer Assisted Intervention, Montreal, QC.

[17] Roget, B., and Sitaraman, J., "Wall Distance Search Algorithm Using Rasterized Marching Spheres", Seventh International Conference on Computational Fluid Dynamics (ICCFD7), Big Island, Hawaii, July 9-13, 2012.

[18] Bonet, J., and Peraire, J., "An Alternating Digital Tree (ADT) Algorithm for 3D geometric searching and intersection problems", International Journal of Numerical Methods in Engineering, Vol 31, 1991, pp.1-17.1

[19] Sitaraman, J., Potsdam, M., Jayaraman, B., Datta, A., Wissink, A., Mavriplis, D. and Saberi, H.,"Rotor Loads Prediction Using HELIOS : A Multi-Solver Framework For Rotorcraft CFD/CSD Analysis" AIAA 2011-1123, 49th AIAA Aerospace Sciences Meeting, 4-7 January 2011, Orlando, Florida.

[20] Wissink, A., Kamkar, S., Pulliam, T. H., Sitaraman, J. and Sankaran, V.,"Cartesian Adaptive Mesh Refinement For Rotorcraft Wake Resolution," Presented at the 67th Forum of American Helicopter Society, Phoenix Arizona, May 2010.

[21] Wissink, A., Jayaraman, B., Potsdam, M., Dimanglig, A. and Lim, J., "Helios Prediction of Blade-Vortex Interaction and Wake of the HART II Rotor," AIAA 2012-714, Presented at the 50th AIAA Aerospace Sciences Meeting, January 2012, Orlando, Florida.

[22] Wissink, A., Jayaraman, B., Datta, A., Sitaraman, J., Potsdam, M., Kamkar, S., Mavriplis, D., Yang, Z., Jain, R., Lim, J., Strawn, R., "Capability Enhancements in Version 3 of the Helios High-Fidelity Rotorcraft Simulation Code", AIAA-2012-713, 50th AIAA Aerospace Sciences Meeting, Nashville TN, Jan 9-12 2012.

[23] Steger, J.L., and Benek, J.A., "On the use of composite grid schemes in computational aerodynamics", Computer Methods in Applied Mechanics and Engineering, Vol.64, No. 1-3, October 1987, pp.301-320.

[24] Suhs, N., Rogers, S., Dietz, W.,“Pegasus 5: An automated pre-processor for overset-grid CFD”, American Institute of Aeronautics and Astronautics 41(6), 10371045 (2003).

[25] Noack, R.W., Boger, D.A., Kunz, R.F., Carrica, P.M., “Suggar++: An improved general overset grid assembly capability”, Proceedings of the 47th AIAA Aerospace Science and Exhibit, Orlando, FL (January 2009).

[26] Alonso, J. J., Hahn, S., Ham, F., Herrmann, M., Iaccarino, G., Kalitzin, G., LeGresley, P., Mattsson, K., Medic, G., Moin, P., Pitsch, H., Schluter, J., Svard, M., Van der Weide, E., You, D., and Wu, X., “CHIMPS: A High-Performance Scalable Module for Multi-Physics Simulations”, 42nd AIAA/ASME/SAE/ASEE Joint Propulsion Conference, AIAA, Washington, DC, July 2006.

[27] Sitaraman, J., Floros, M., Wissink, A., Potsdam, M. ”Parallel Domain Connectivity Algorithm For Unsteady Flow Computations Using Overlapping And Adaptive Grids,” Journal of Computational Physics, Volume 229, Issue 12, p. 4703-4723.

[28] Meakin, R.L., “Object X-Rays for Cutting Holes in Composite Overset Structured Grids,” 15th AIAA Computational Fluid Dynamics Conference, AIAA, Washington, DC, June 2001.

[29] Buning, P.G. and Pulliam, T.H., “Cartesian Off-body Adaptation For Viscous Time-Accurate Flow Simulations”, AIAA Paper 2011-3693, 20th AIAA Computational Fluid Dynamics Conference, June 27-30, 2011, Honolulu, Hawaii.

[30] Belk, D. M. and Maple, R. C., “Automated Assembly of Structured Grids for Moving Body Problems”, 12th AIAA Computational Fluid Dynamics Conference, Part 1, AIAA, Washington, DC, June 1995, pp. 381390.

[31] Wang, Z. J., Parthasarathy, V., and Hariharan, N., “A Fully Automated Chimera Methodology for Multiple Moving Body Problems”, 36th AIAA Aerospace Sciences Meeting, AIAA, Washington, DC, January 1998.

[32] David L. Brown, W. D. H. and Quinlan, D. J., “Overture: Object-Oriented Tools for Overset Grid Applications”, 17th AIAA Conference on Applied Aerodynamics, AIAA, Washington, DC, June 1999.

[33] JeanFaivre, G., Juvigny, X., and Benoit, C., “Parallel Chimera Computations of Helicopter Flows”, 24th International Congress of the Aeronautical Sciences, ICAS 2004.

[34] Sankaran, V., Sitaraman, J., Wissink, A., Datta, A., Jayaraman, B., Potsdam, M., Mavriplis, D., Yang, Z., O’Brien, D., Saberi, H., Cheng, R., Hariharan, N., and Strawn, R., “Application of the Helios computational platform to rotorcraft Flowfields”, 48th AIAA Aerospace Sciences Meeting Including the New Horizons forum and Aerospace Exposition, 4-7 January 2010, Orlando, Florida.

[35] Post, D.E., “A new Dom initiative: the Computational Research and Engineering Acquisition Tools and Environments (CREATE) program”, Journal of Physics, Conference Series 125, 2008.

[36] Zagaris, G., Campbell, M.T., Bodony, D.J., Shaffer, E. and Brandyberry, M.D., “A Toolkit for Parallel Overset Grid Assembly Targeting Large-Scale Moving Body Aerodynamic Simulation”, Sandia National Laboratories, Proceedings of the 19th International Meshing Roundtable, 2010, pp.385-401.

[37] Pissanetzky, S. and Basombrio, F.G., "Efficient Calculation of Numerical Values of a Polyhedral Function", International Journal of Numerical Methods in Engineering, Vol.17, 1981, pp.231-237.

[38] Khoshniat, M., Stuhne, G.R., and Steinman, D.A., "Relative Performance of Geometric Search Algorithms for Interpolating Unstructured Mesh Data", MICCAI 2003, 6th International Conference on Medical Image Computing and Computer Assisted Intervention, Montreal, QC.

[39] Roget, B., and Sitaraman, J., "Wall Distance Search Algorithm Using Rasterized Marching Spheres", Seventh International Conference on Computational Fluid Dynamics (ICCFD7), Big Island, Hawaii, July 9-13, 2012.

[40] Bonet, J., and Peraire, J., "An Alternating Digital Tree (ADT) Algorithm for 3D geometric searching and intersection problems", International Journal of Numerical Methods in Engineering, Vol 31, 1991, pp.1-17.

[41] van der Wall, B.G., Burley, C.L., Yu, Y., Richard H., Pengel K., and Beaumier, P., "The HART II Test - Measurement of Helicopter Rotor Wakes" Aerospace Science and Technology 8(2004):273-284.

[42] Morton, S.A., Lamberson, S.E., and McDaniel, D.R., "Static and Dynamic Aeroelastic Simulations using *Kestrel* - a CREATE Aircraft Simulation Tool", 53rd AIAA Structures, Structural Dynamics and Materials Conference, 23-26 April 2012, Honolulu, Hawaii.

[43] Message Passing Interface *http://www.mcs.anl.gov/research/projects/mpich2/*

[44] Python Programming Language – Official Website *http://www.python.org/*

[45] Numerical python *http://www.numpy.org/*

[46] P. Peterson F2PY: Fortran to Python interface generator *http://cens.ioc.ee/projects/f2py2e/*

[47] L. Dalcin MPI for Python *http://mpi4py.scipy.org/*